Cloud Native Computing:

Orchestrating Containers For Better Service and Application Deployments

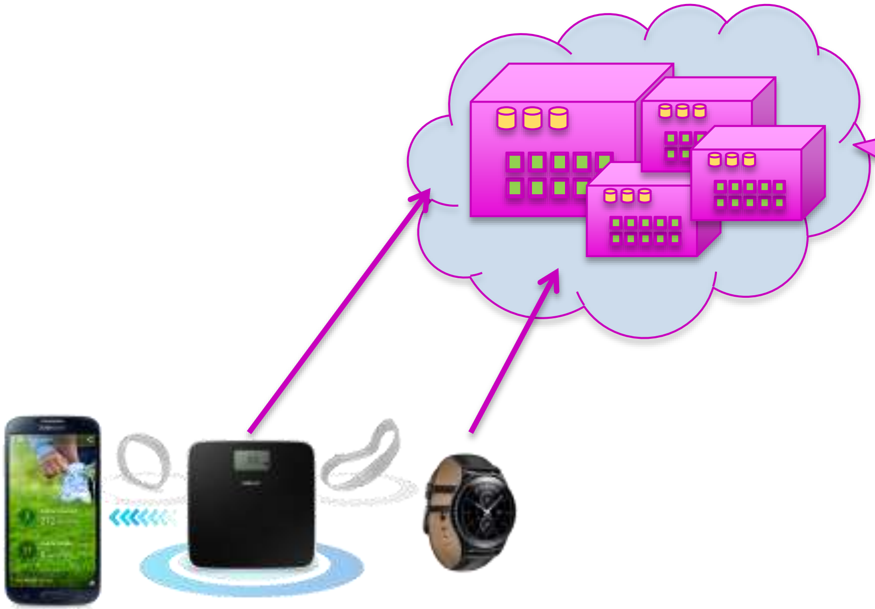
Richard Kaufmann, VP
Bob Wise, CTO
Cloud Native Computing Team
Samsung SDS Research America

SAMSUNG
DEVELOPER
CONFERENCE™

This presentation is intended to provide information concerning Samsung’s efforts around containers and container orchestration. We do our best to make sure that information presented is accurate and fully up-to-date. However, the presentation may be subject to technical inaccuracies, information that is not up-to-date or typographical errors. As a consequence, Samsung does not in any way guarantee the accuracy or completeness of information provided on this presentation. Samsung reserves the right to make improvements, corrections and/or changes to this presentation at any time.

The information in this presentation or accompanying oral statements may include forward-looking statements. These forward-looking statements include all matters that are not historical facts, statements regarding Samsung Data System' intentions, beliefs or current expectations concerning, among other things, market prospects, growth, strategies, and the industry in which Samsung operates. By their nature, forward-looking statements involve risks and uncertainties, because they relate to events and depend on circumstances that may or may not occur in the future. Samsung cautions you that forward looking statements are not guarantees of future performance and that the actual developments of Samsung, the market, or industry in which Samsung operates may differ materially from those made or suggested by the forward-looking statements contained in this presentation or in the accompanying oral statements. In addition, even if the information contained herein or the oral statements are shown to be accurate, those developments may not be indicative developments in future periods.

Open source projects and 3rd party products are mentioned for informational purposes only. No endorsement is expressed or implied. Logos and trademarks remain the property of their respective owners.



This Talk: New software to “orchestrate” the back-end of modern mobile & IoT services

Monolithic	Client-Server	N-tier		Microservices
Proprietary Mainframe	Open Systems	Virtualization	Cloud	Container-centric
Operators & Runbooks		Automation		Orchestration

Characterizing the Gap

Old School

Self-managed basic IT

Monolithic
Complex Dependencies

Gold Plated, Reliable Hardware
Expensive
Proprietary
Design to Prevent Failure

Large, Centrally Controlled Teams
Months to Production
Waterfall Process
Ponderous, Manual QA
Quality Issues
Costly

Cloud Native

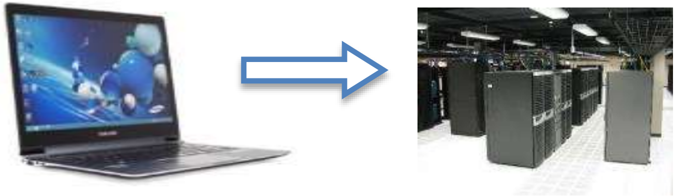
SaaS for basic IT (Email, etc)

Microservices
Lots of Simple, Independent Parts

Commodity Hardware
Software Reliability
Open Source
Designed to Fail
Chaos Monkey

Small, Independent Teams
Continuous Integration
Continuous Deployment
Changes Made in Minutes
High Quality
Ubiquitous Automation
Efficient

Driving Force: Containers



- Complex, distributed services built on notebooks, which then run, unmodified, on public or private infrastructure
- Isolation from dependencies & from each other
- Highly efficient (low memory footprint, low startup time, rapid scale up/down – often 10X overall improvement)

High Bar - Industrial Grade Compute
as practiced by Google, Facebook, Amazon, Twitter, …

- 70% Datacenter utilization
- 10,000-40,000 servers per admin
- Commodity hardware
- Extremely low licensing spend
- Comprehensive Automation
- Extensive Analytics
- Multiple releases a day / CD
- Destructive testing in production (chaos monkey)
- No-ops (Self service deployment)

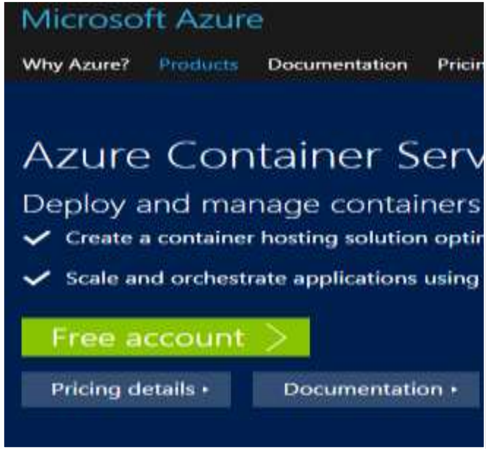
CLOUD NATIVE COMPUTING FOUNDATION

#GIFEE

Cisco
CoreOS
docker
Google
HUAWEI
IBM
intel
Joyent
MESOSPHERE
redhat
SUPERNAP
Goldman Sachs
at&t
NetApp
APCERA
APPFORMIX
apprenda
cloudsoft
ClusterHQ
Container Solutions
Datawise.io
DEIS
ebay
eldarion
FUJITSU
iguaz.io
Infoblox
ishi systems
KISMATIC
CALICO
NCSOFT
NEC
NGINX
portworx
RANCHER
RX-M
Scalock
sysdig
Twistlock
box
UNIVA
Virtuozzo
vmware
weaveworks
WILLIAMS GARCIA
Centrify

Service Providers

Kubernetes-powered


Microsoft Azure

Why Azure? Products Documentation Pricin

Azure Container Serv

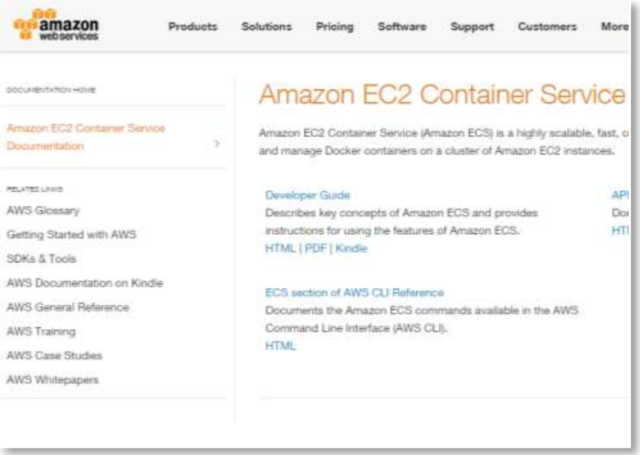
Deploy and manage containers

✓ Create a container hosting solution opti

✓ Scale and orchestrate applications using

Free account >

Pricing details ▸ Documentation ▸


amazon web services

Products Solutions Pricing Software Support Customers More

DOCUMENTATION HOME

Amazon EC2 Container Service Documentation

RELATED LINKS

AWS Glossary

Getting Started with AWS

SDKs & Tools

AWS Documentation on Kindle

AWS General Reference

AWS Training

AWS Case Studies

AWS Whitepapers

Amazon EC2 Container Service

Amazon EC2 Container Service (Amazon ECS) is a highly scalable, fast, c and manage Docker containers on a cluster of Amazon EC2 instances.

Developer Guide

Describes key concepts of Amazon ECS and provides instructions for using the features of Amazon ECS.

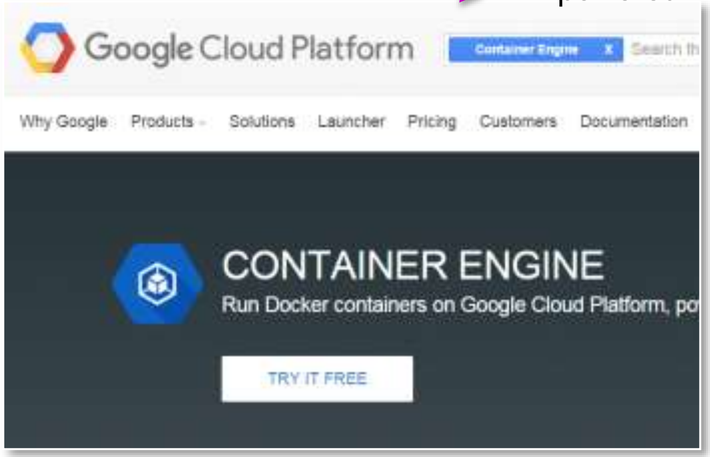
HTML | PDF | Kindle

API Doc HTI

ECS section of AWS CLI Reference

Documents the Amazon ECS commands available in the AWS Command Line Interface (AWS CLI).

HTML


Google Cloud Platform

Container Engine X Search th

Why Google Products Solutions Launcher Pricing Customers Documentation

CONTAINER ENGINE

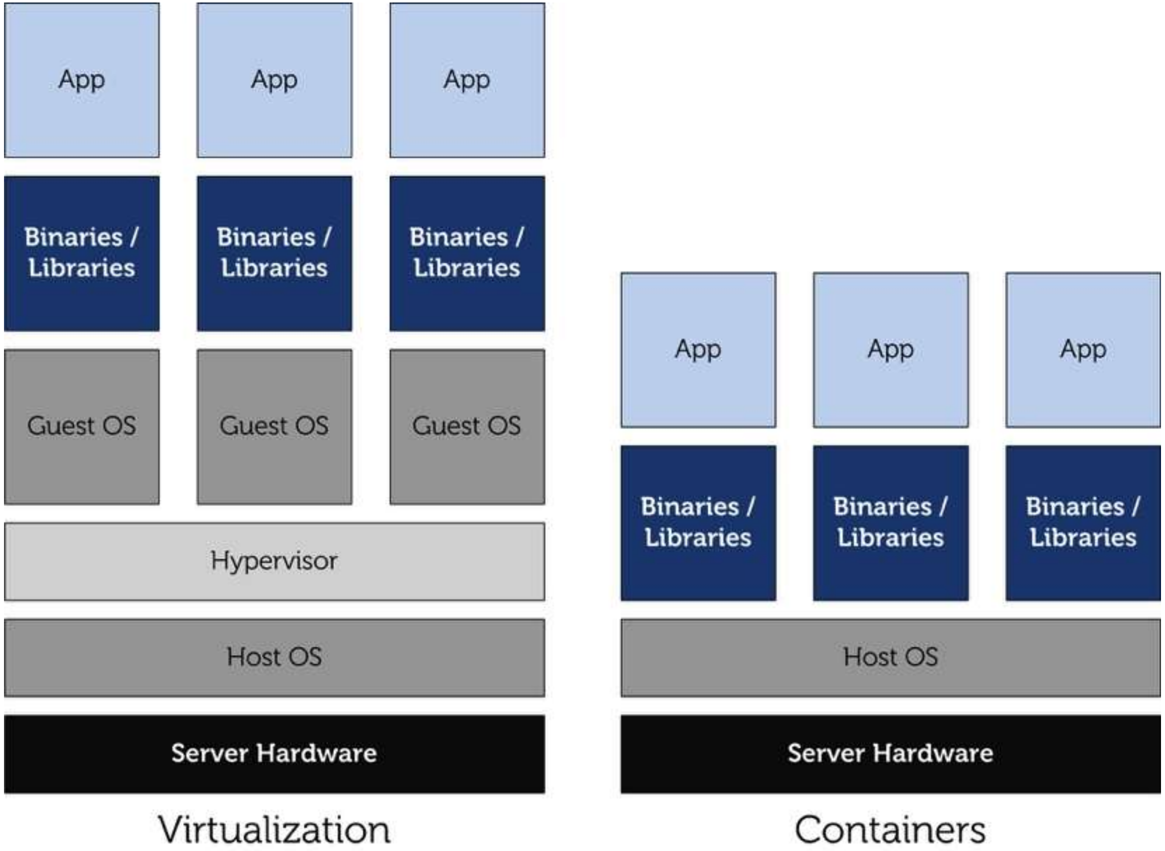
Run Docker containers on Google Cloud Platform, po

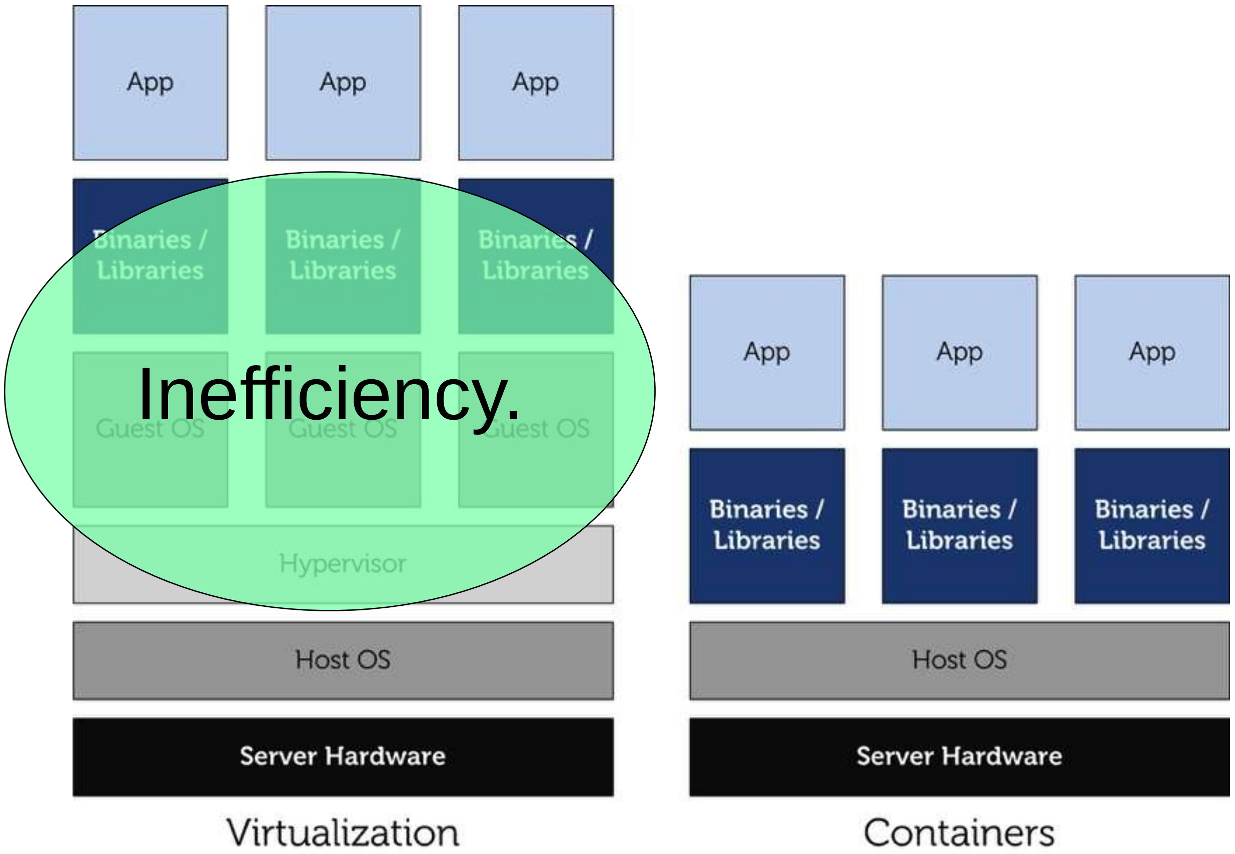
TRY IT FREE

Quick refresher…
Container and Docker 101

SAMSUNG DEVELOPER CONFERENCE™

Containers and VMs



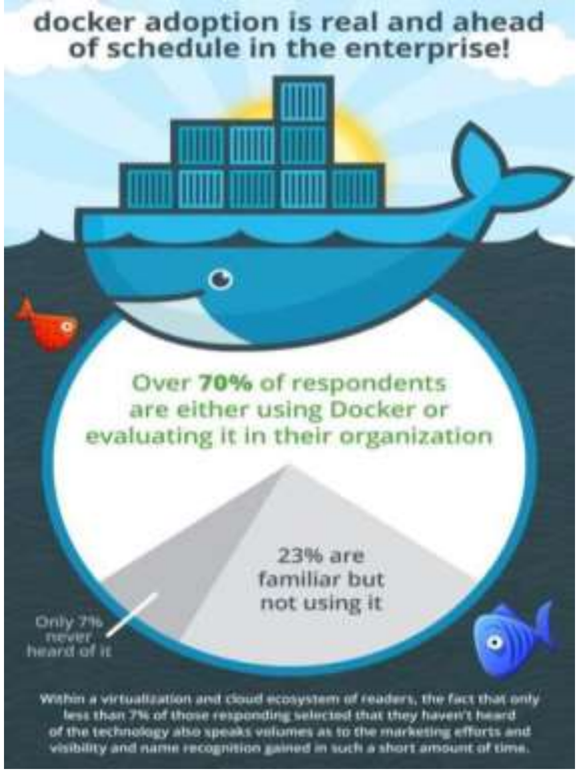


Inefficiency.

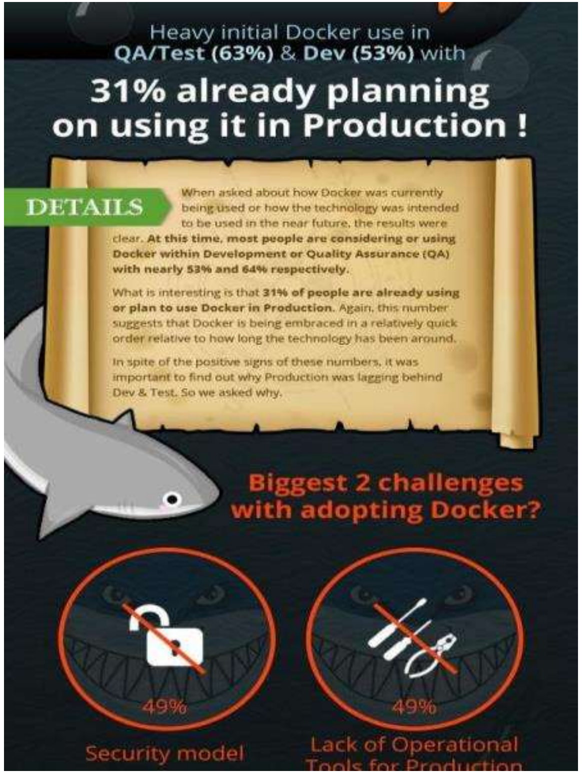
Human Efficiency: Bigger Gains

- Docker’s Breakthrough: Good tools and standardization built on top of mature container technology
- Method for deploying “anywhere” by ensuring all the dependencies get built into the container with the application
  - Better CI/CD
  - More automation in operations
  - More self-service and dev/ops
- Open source model → Collaboration → Velocity
  - Open repositories of images to start from… you can stand on the shoulders of giants
- Enables application designs that are better for humans to manage

Containers are mainstream…


docker adoption is real and ahead of schedule in the enterprise!
Over 70% of respondents are either using Docker or evaluating it in their organization
23% are familiar but not using it
Only 7% never heard of it

Within a virtualization and cloud ecosystem of readers, the fact that only less than 7% of those responding selected that they haven't heard of the technology also speaks volumes as to the marketing efforts and visibility and name recognition gained in such a short amount of time.


Heavy initial Docker use in QA/Test (63%) & Dev (53%) with
31% already planning on using it in Production !
DETAILS
Biggest 2 challenges with adopting Docker?
49%
Security model
49%
Lack of Operational Tools for Production

When asked about how Docker was currently being used or how the technology was intended to be used in the near future, the results were clear. **At this time, most people are considering or using Docker within Development or Quality Assurance (QA) with nearly 53% and 64% respectively.**

What is interesting is that **31% of people are already using or plan to use Docker in Production**. Again, this number suggests that Docker is being embraced in a relatively quick order relative to how long the technology has been around.

In spite of the positive signs of these numbers, it was important to find out why Production was lagging behind Dev & Test. So we asked why.

www.docker.com

- Usage moving from the big service providers (Google, Twitter, etc.) to enterprises.
- Even very traditional enterprises are adopting the new technology - Cisco IT using Red Hat Openshift, Kubernetes and Containers for production workloads.

Disruptive Effects on Classic IT

- **Low Entry Barrier**
  - Serverless Startups
  - ShadowIT
- **Deprecates existing technologies**
  - Shared block storage (SANs)
  - Shared filesystems (NAS)
  - Legacy networking, e.g. h/w firewalls
  - Legacy “fat” Linux distros
- **Less Labor (Tinier Empire!!!)**
  - Automated deployment, scaling, repair, teardown
- **Cheaper servers and network h/w**
  - Strictly regimented, homogenous
  - Cattle, not pets.
- **Key technologies open sourced**

CLOUD NATIVE
COMPUTING FOUNDATION

It’s NOT just Containers!

1. **Containers**: Benefits of VMs, but lighter weight, portable across laptops → massive scale deployment.

2. **Dynamically managed**: Services start, stop, scale up/down and are repaired automatically.

3. **Microservices**: Built from loosely-coupled, independently testable pieces – significantly increases the agility, scalability and maintainability of applications.

→ Faster to develop and deploy
→ Easier to scale (and autoscale!)
→ More efficient to operate
→ Much higher service quality
→ Far fewer outages

www.cncf.io

What’s a Microservice???

- Not: 10-100 lines of code

- Loosely coupled architecture
  - Accessed via APIs (typically REST or http)
- Pieces independently…
  - Upgradable
  - Scalable
  - Developed (small teams)

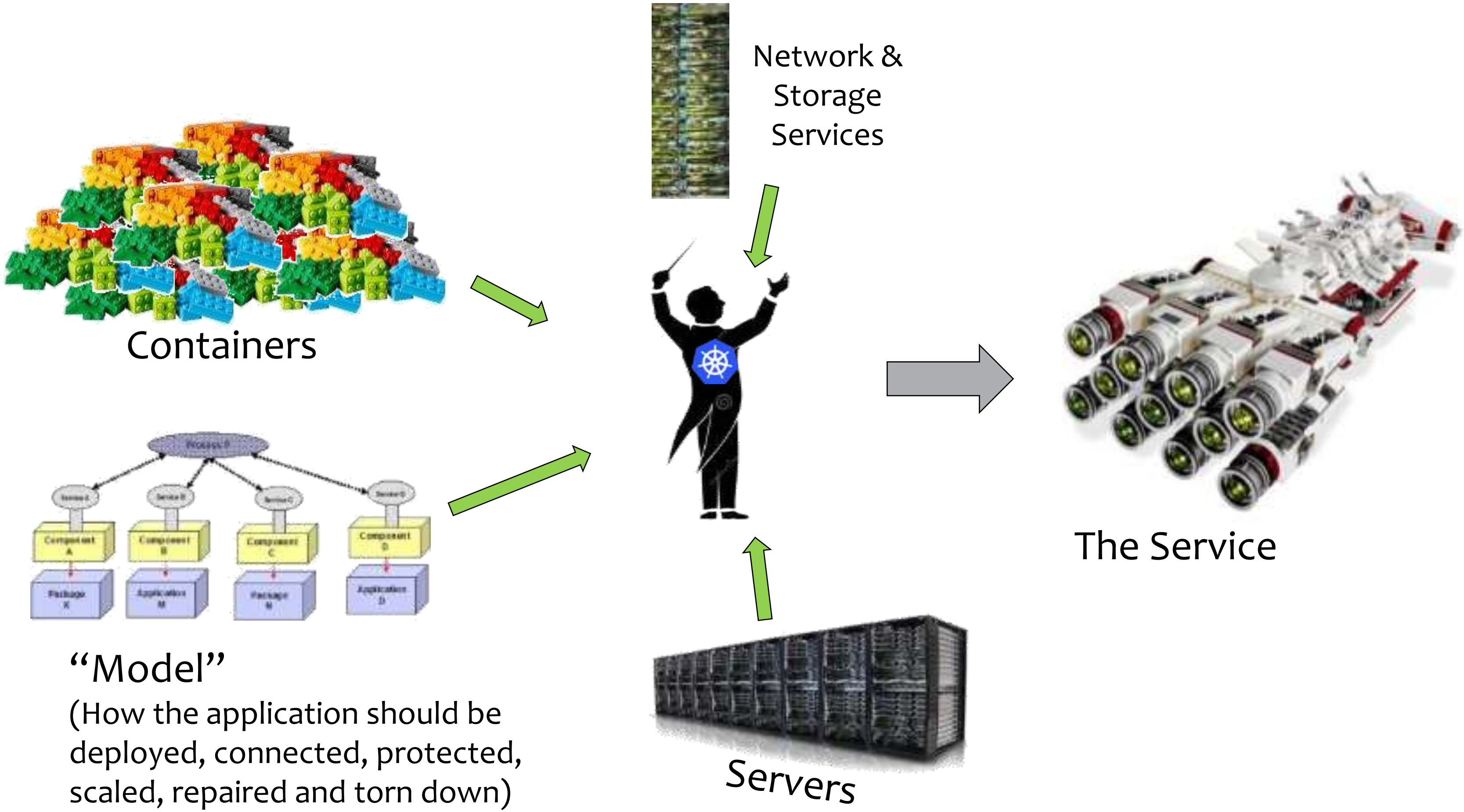
Org Impacts

- Microservices = Reorg
- Ops Automation + Specialization = Reorg
- High premium on technically proficient product management
- High premium on operations engineering, pipeline development, and internal platforms

Putting It All Together

- Your app or service
  - Pods (groups of containers on the same server image)
  - A description of your service: “*the model*” describing the pods, storage & network needs, health metrics, and scale up/down directives
- A cluster
  - Made up of a number of *nodes* (virtual or physical servers)
  - Connected by a physical network to each other and the internet
- A networking technology
  - An SDN or router technology to allow each service to operate independently of the others
  - Load balancers, …
- A “thin host”
  - CoreOS, Snappy, Atomic, Photon, …
- And an orchestration technology to glue everything together

Kubernetes: The leading open source Cloud Native orchestration technology



Network & Storage Services

Containers

“Model”
(How the application should be deployed, connected, protected, scaled, repaired and torn down)

Servers

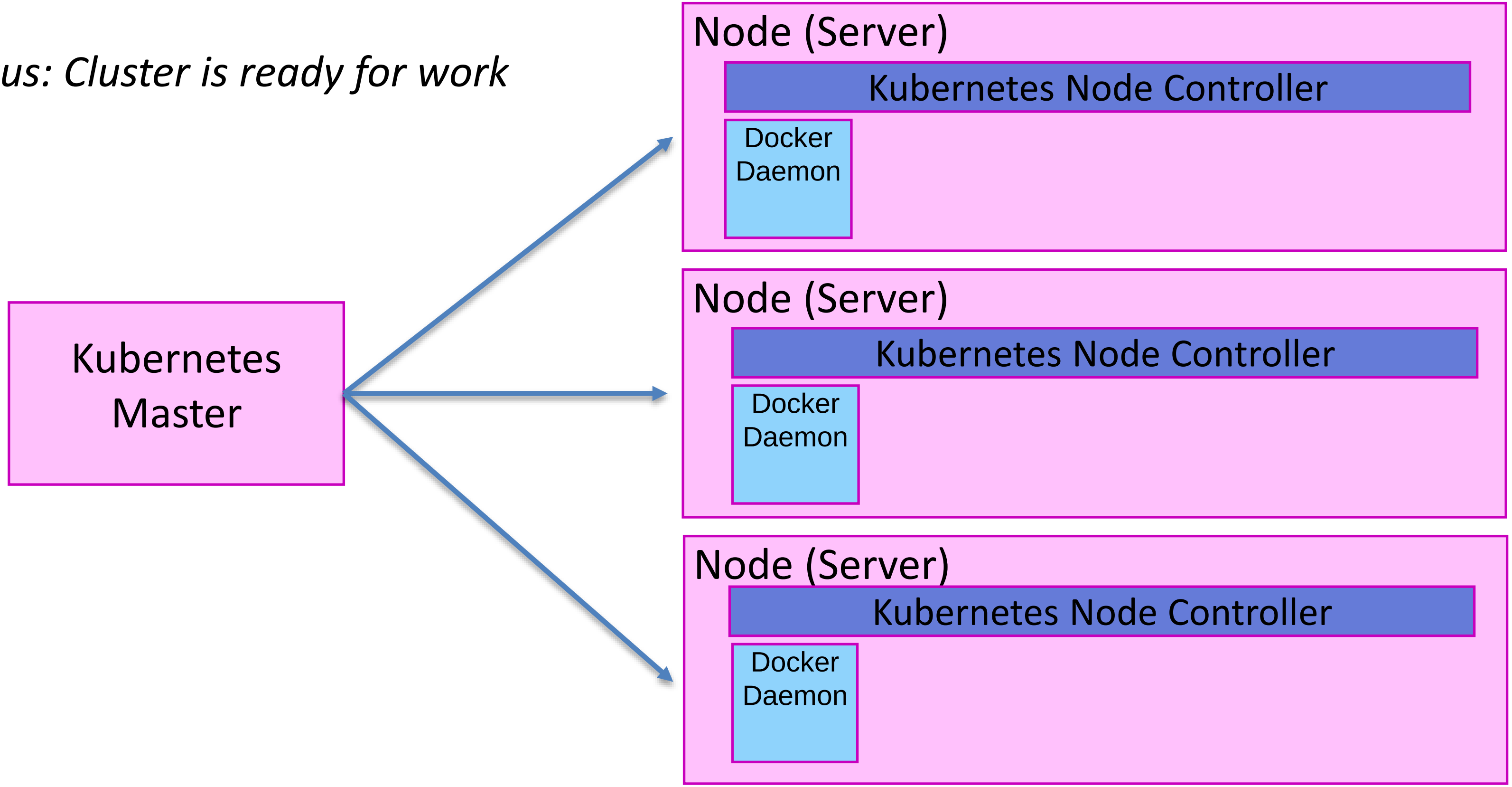
The Service

Kubernetes 101

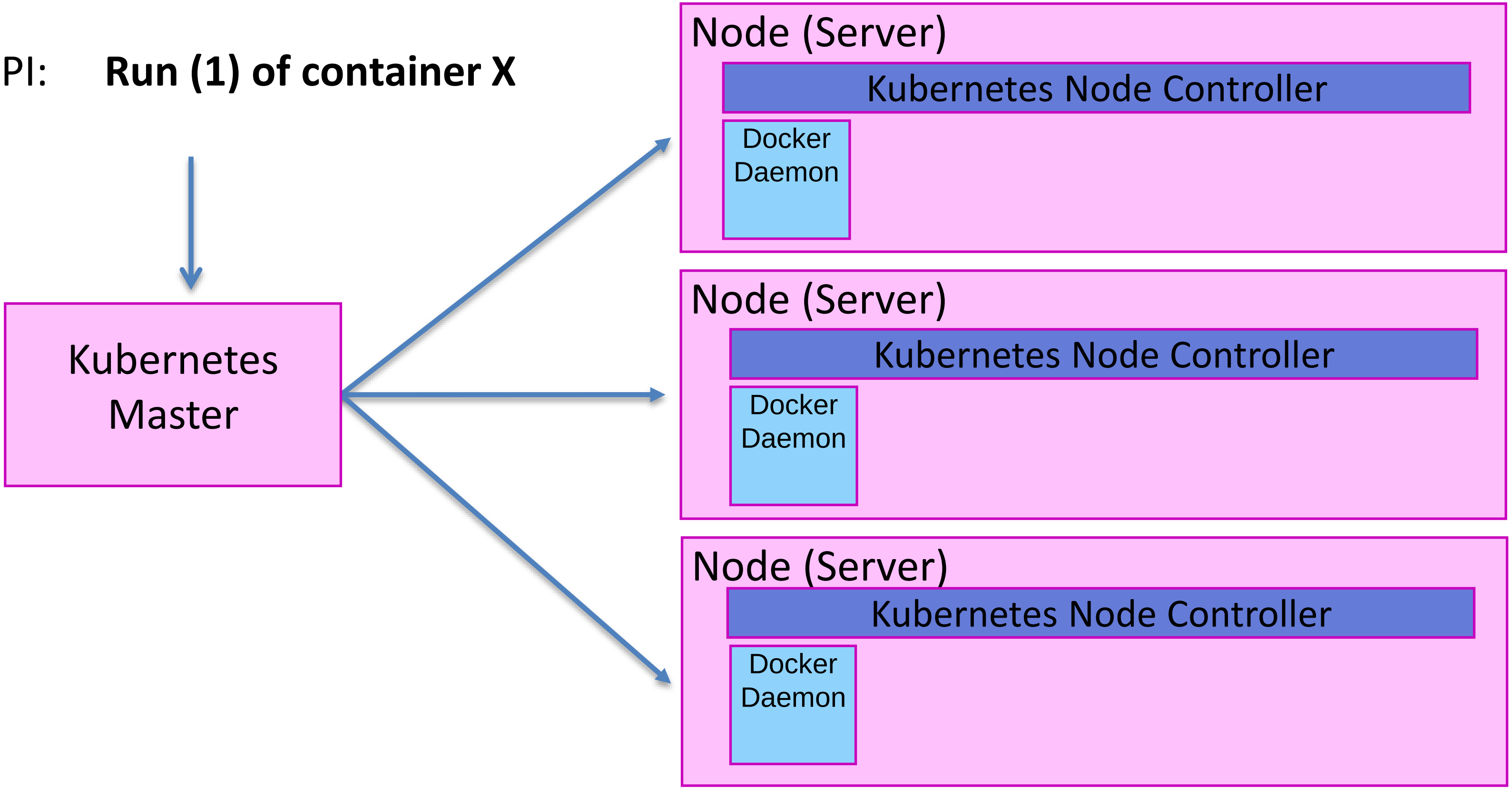
www.kubernetes.io

Kubernetes Conceptual

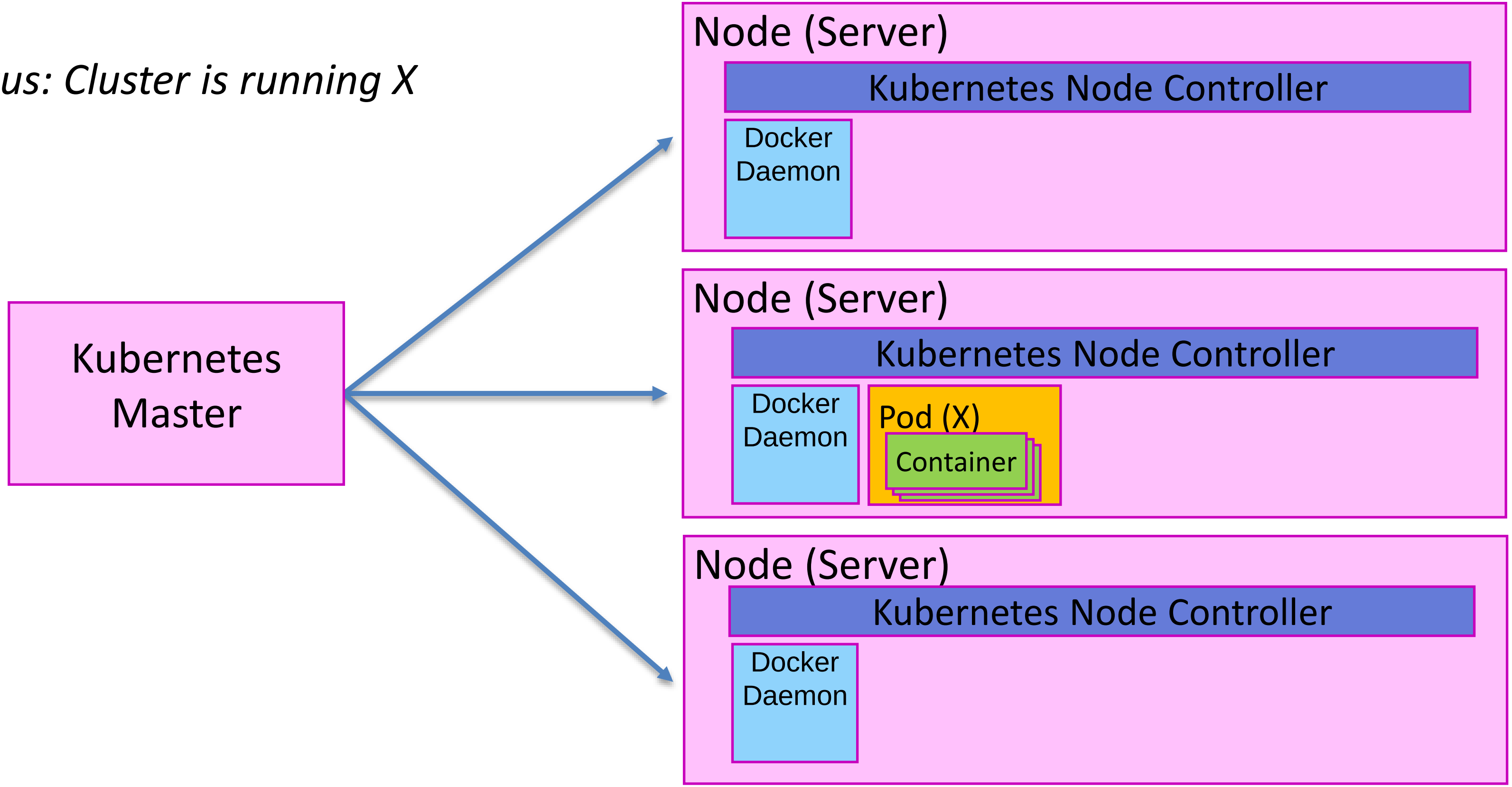
Status: Cluster is ready for work



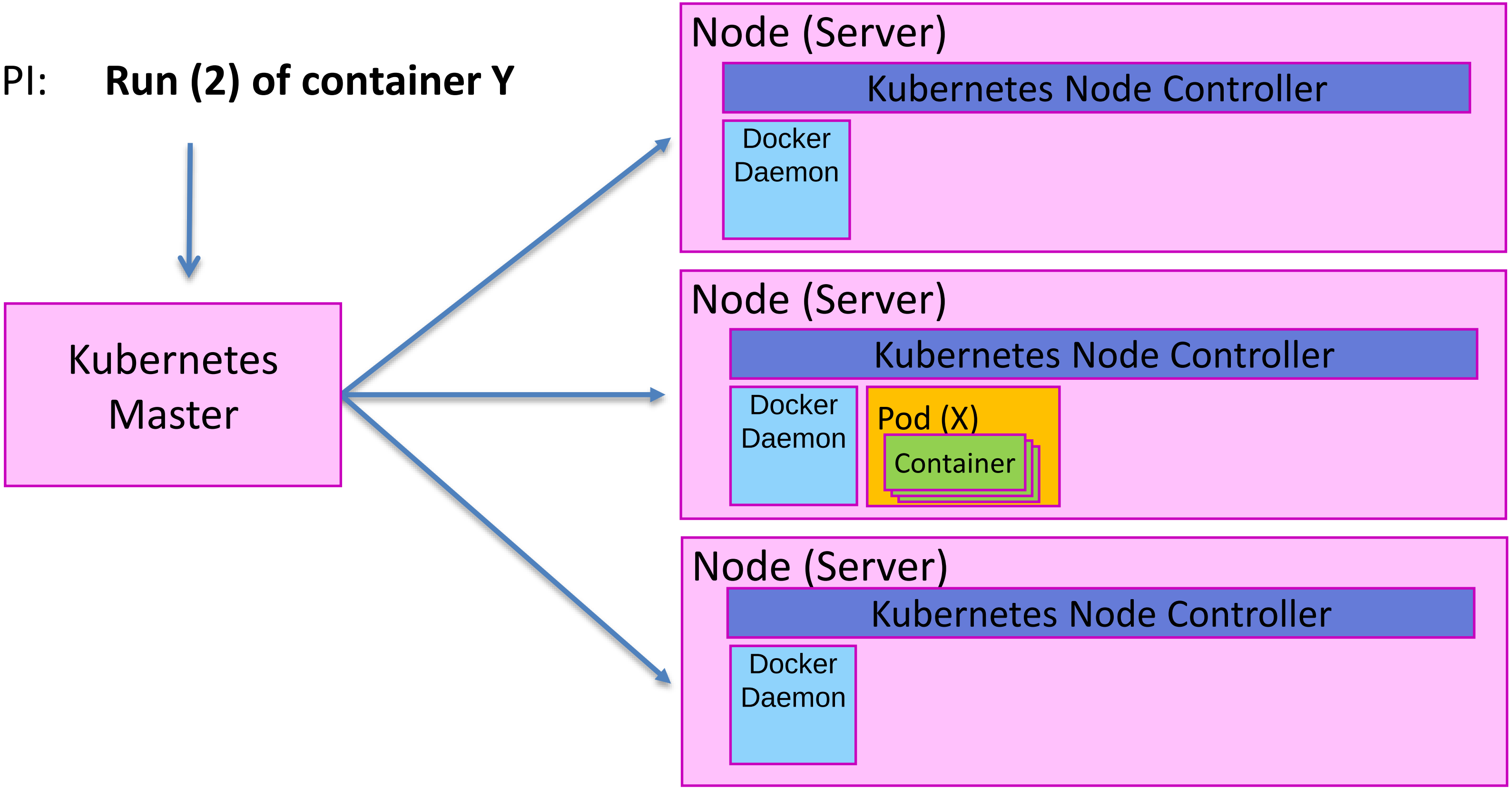
To API: **Run (1) of container X**



Status: Cluster is running X



To API: **Run (2) of container Y**



Kubernetes Master

Node (Server)

Kubernetes Node Controller

Docker Daemon

Node (Server)

Kubernetes Node Controller

Docker Daemon

Pod (X)

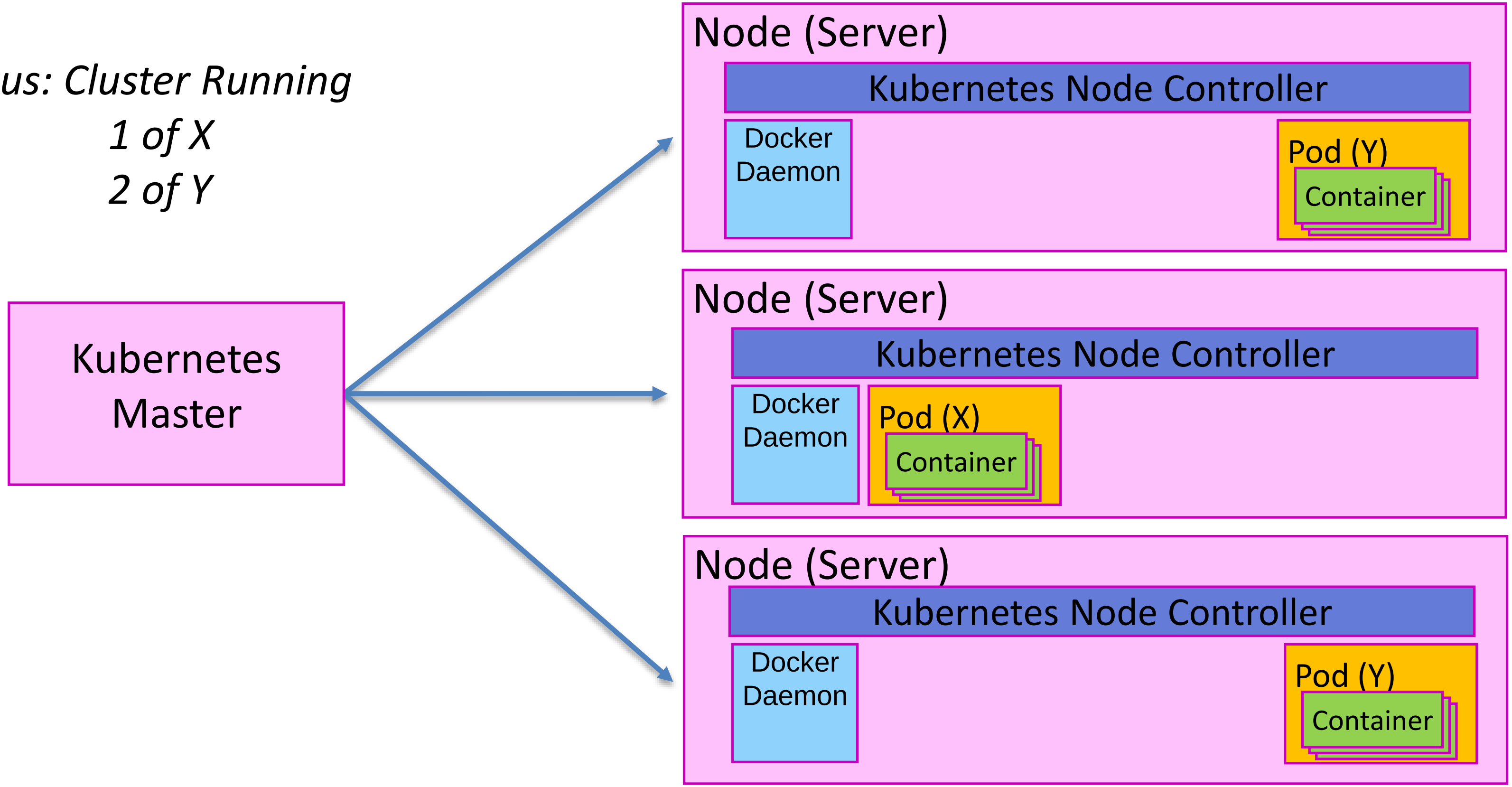
Container

Node (Server)

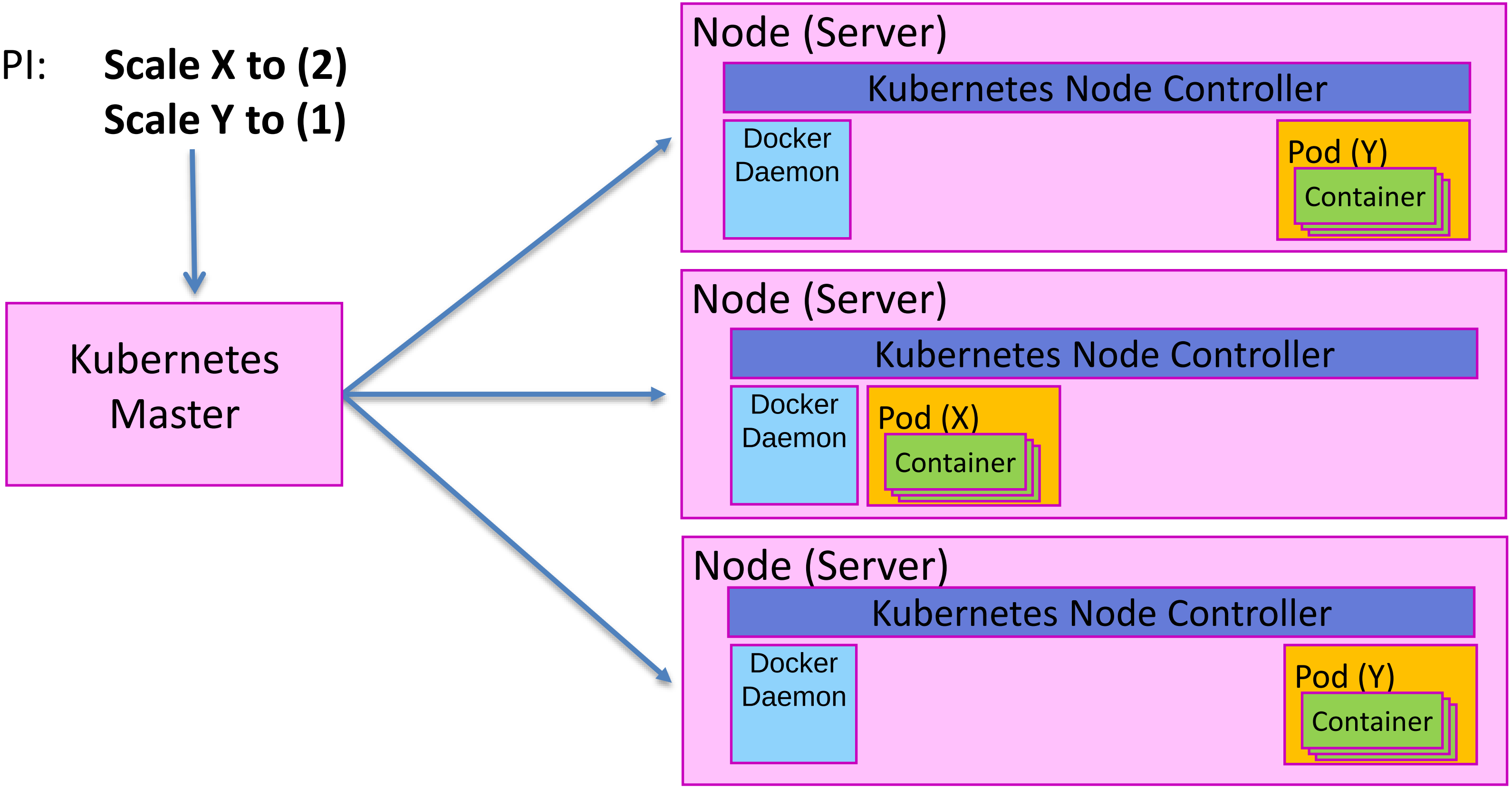
Kubernetes Node Controller

Docker Daemon

Status: Cluster Running
1 of X
2 of Y



To API: **Scale X to (2)**
Scale Y to (1)



Kubernetes Master

Node (Server)

Kubernetes Node Controller

Docker Daemon

Pod (Y)

Container

Node (Server)

Kubernetes Node Controller

Docker Daemon

Pod (X)

Container

Node (Server)

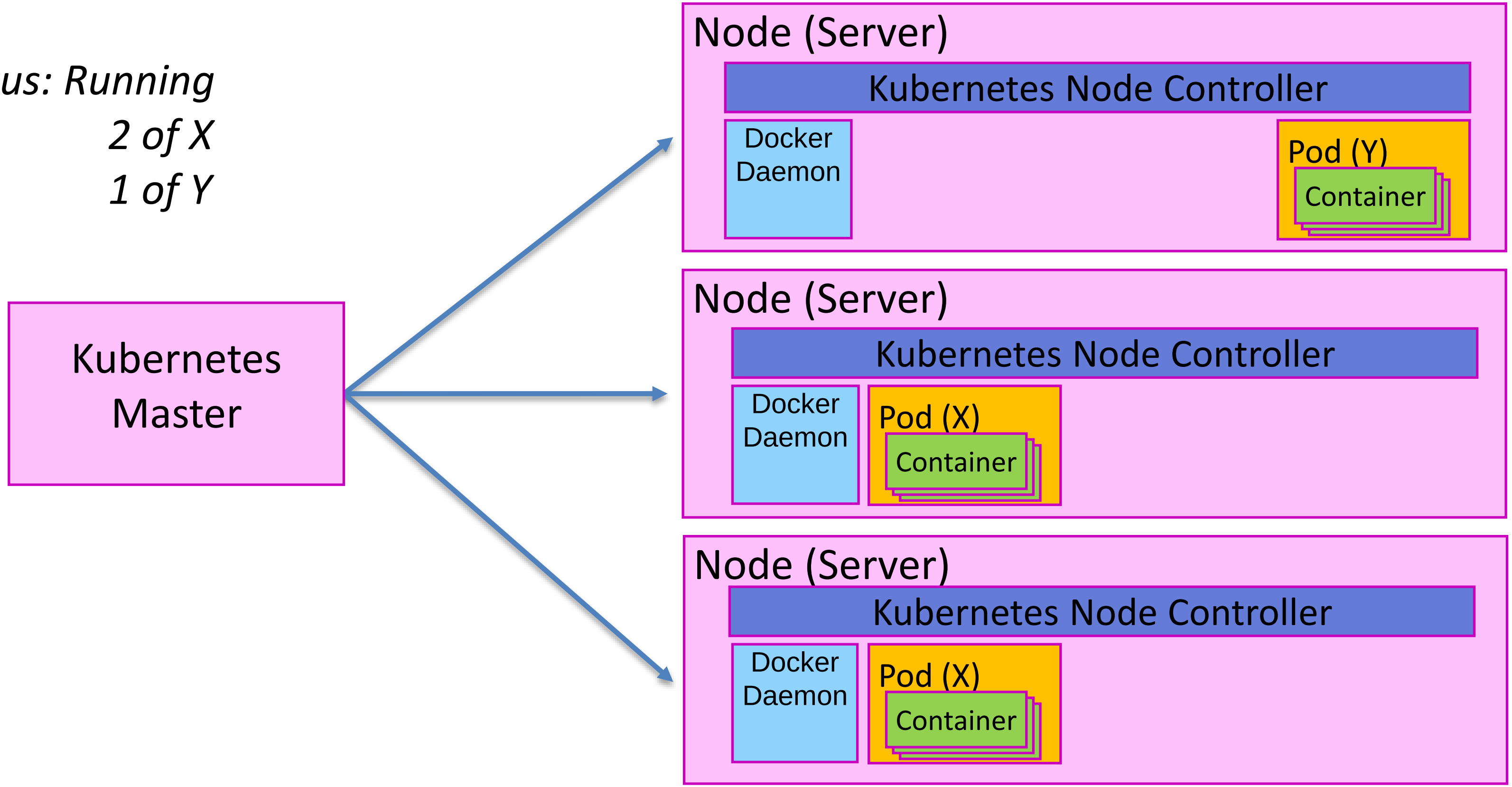
Kubernetes Node Controller

Docker Daemon

Pod (Y)

Container

Status: Running
2 of X
1 of Y



Kubernetes Master

Node (Server)

Kubernetes Node Controller

Docker Daemon

Pod (Y)

Container

Node (Server)

Kubernetes Node Controller

Docker Daemon

Pod (X)

Container

Node (Server)

Kubernetes Node Controller

Docker Daemon

Pod (X)

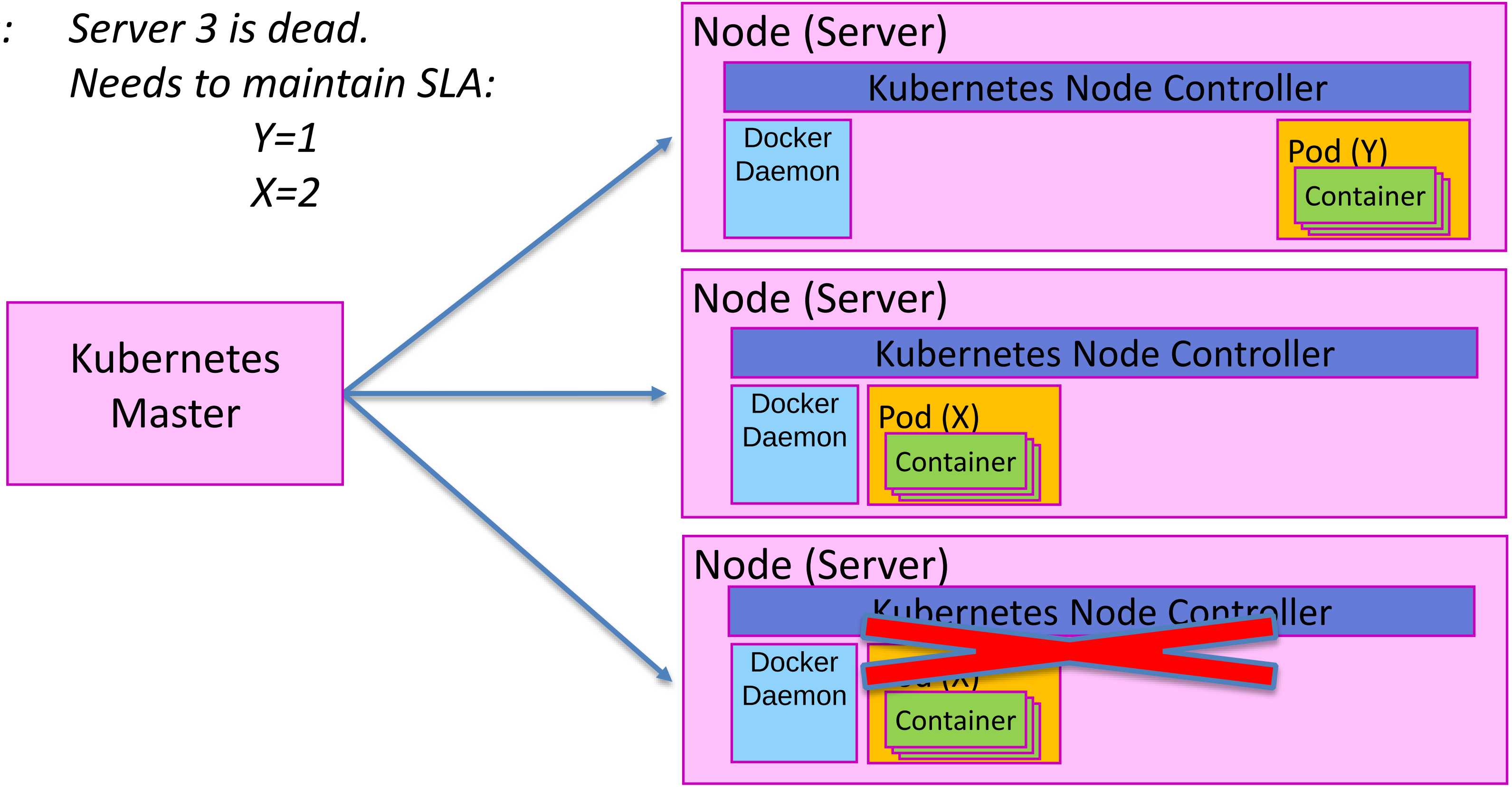
Container

Status: Server 3 is dead.

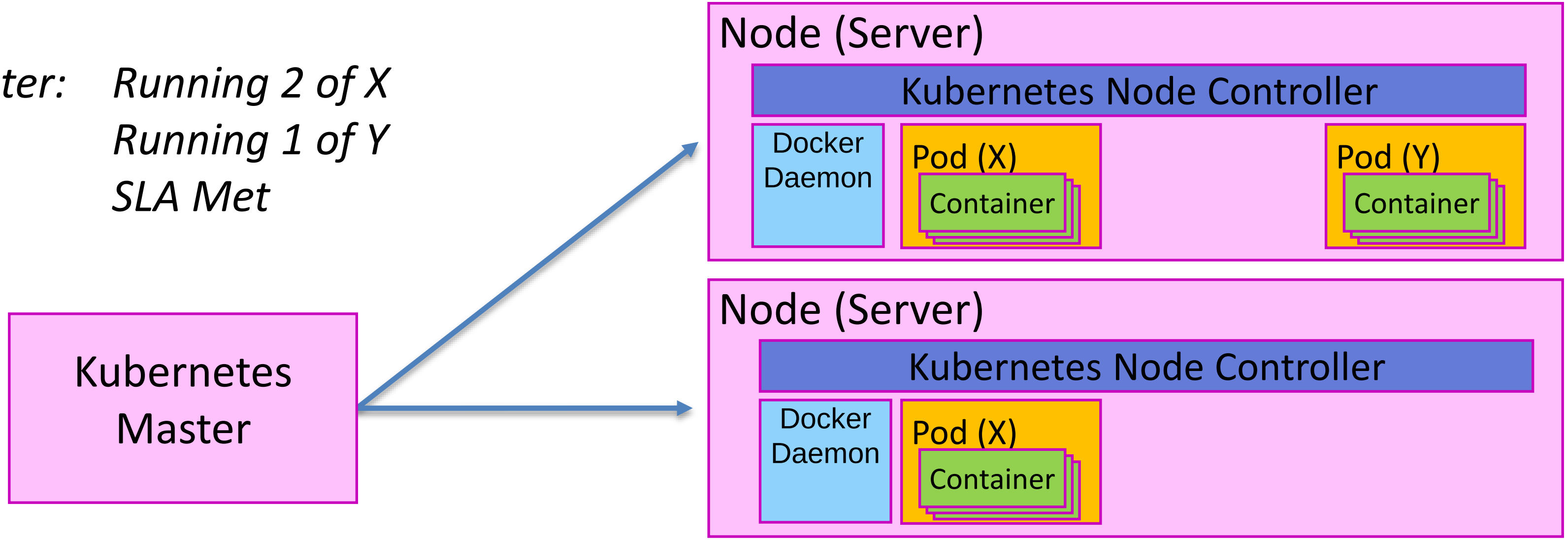
Needs to maintain SLA:

Y=1

X=2



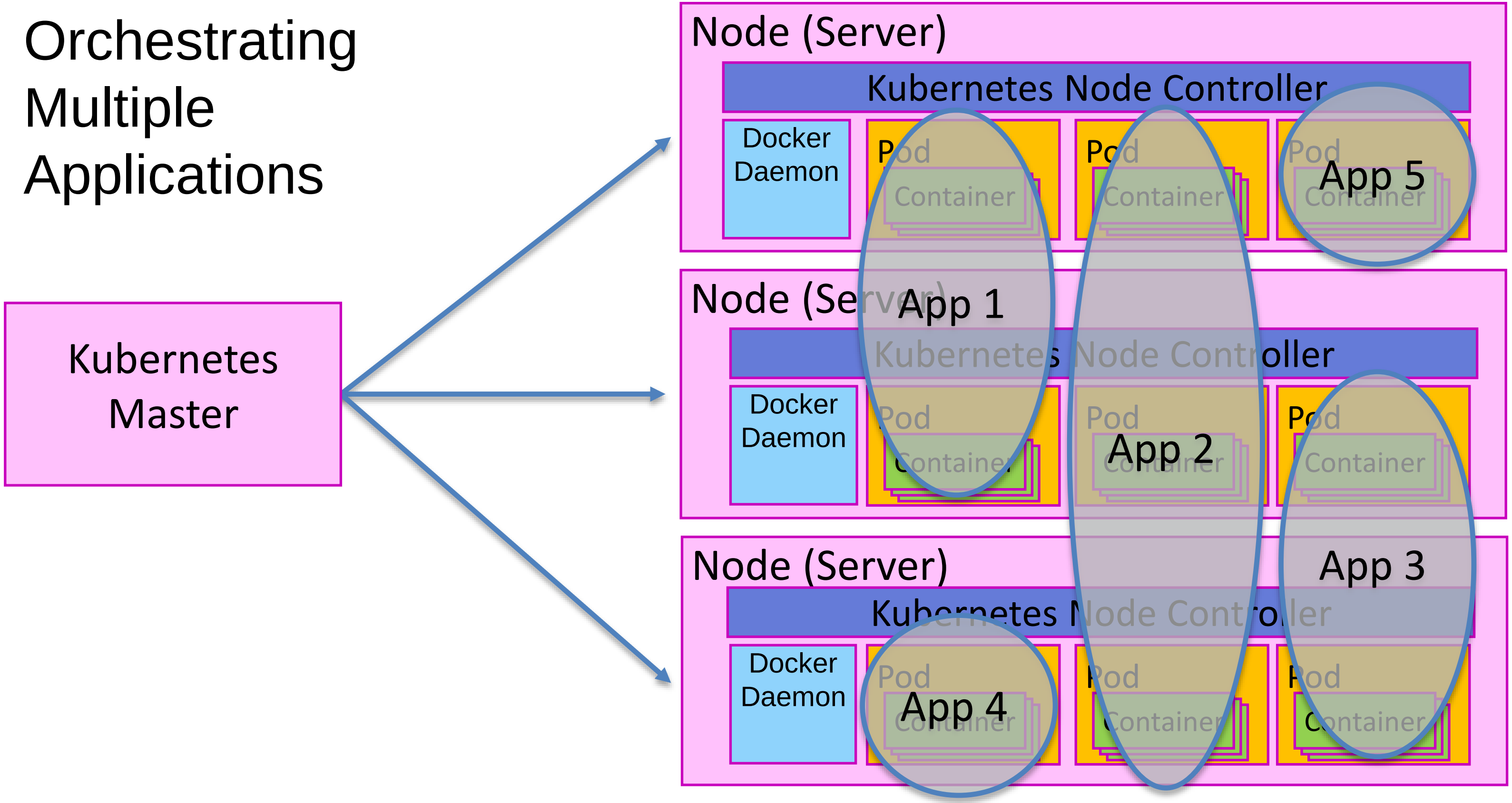
Cluster: Running 2 of X
Running 1 of Y
SLA Met



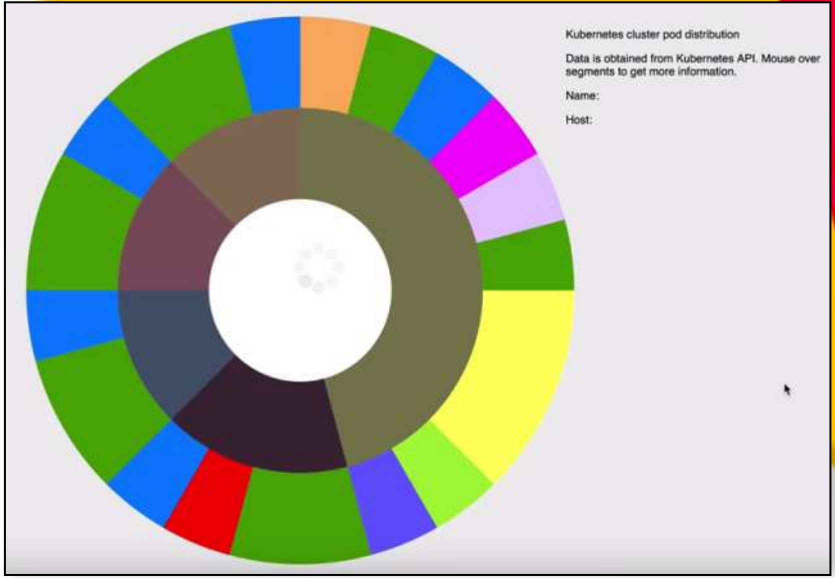
Kubernetes is a “declarative system”...

- Declare what you need “Make sure 2 of X is running”
- The system works to achieve it automatically
- This is zero-touch operations automation

Orchestrating Multiple Applications



Demo



Kubernetes cluster pod distribution

Data is obtained from Kubernetes API. Mouse over segments to get more information.

Name:

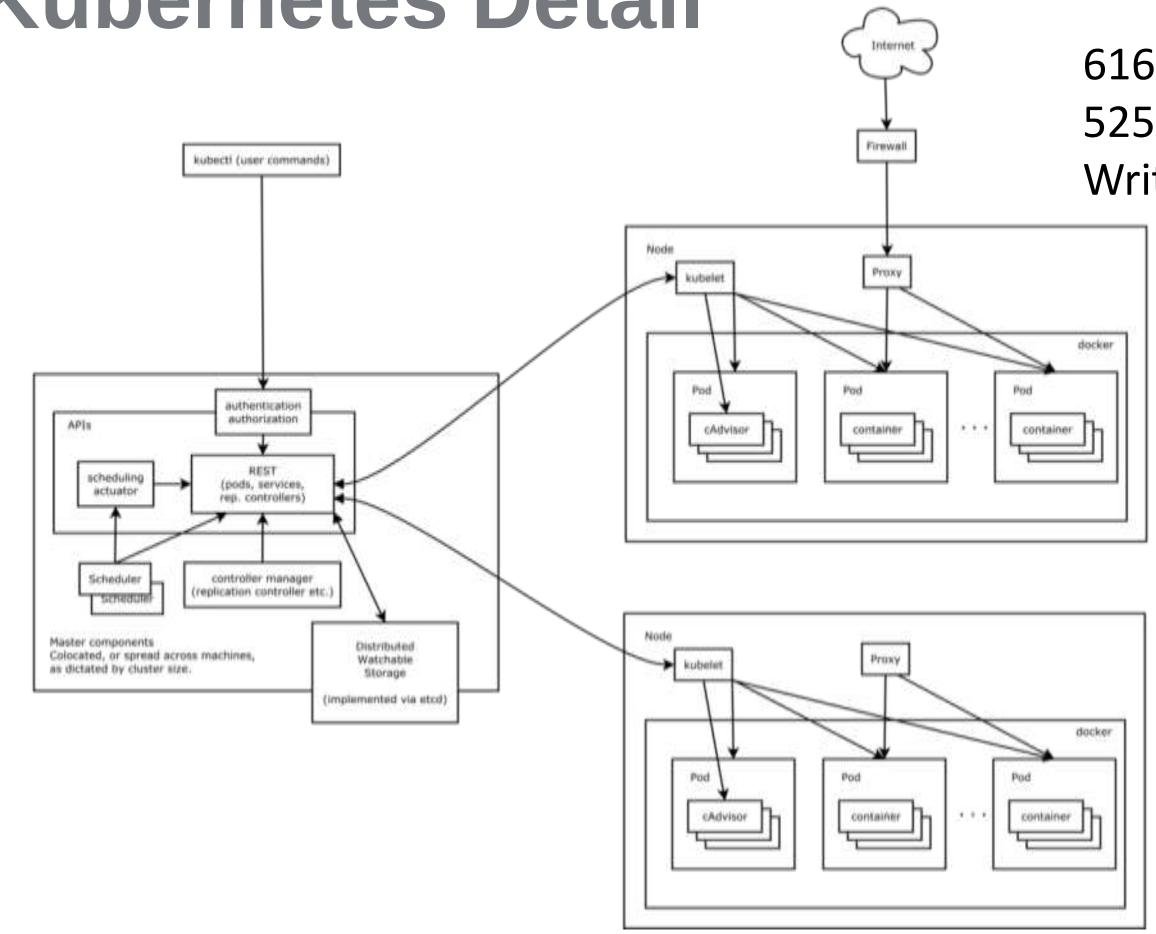
Host:

Kubernetes is for apps & datacenters

- Type I (Dedicated Cluster for an App)
  - Kubernetes manages services and applications
  - These run anywhere…
    - Amazon, Google Compute Engine, your own datacenter
    - Or a “Container Service” (e.g. GKE, Google’s Kubernetes-fueled container engine)
  - See www.github.com/samsung-cnct for a way to configure Kubernetes on AWS (GCE and bare metal in progress)

- Type II (Shared Cluster running many Apps)
  - Kubernetes manages the datacenter
  - It can host your services and apps
  - But can also manage resources supplied to others
    - OpenStack (sub-)clusters
    - Etc.

Kubernetes Detail



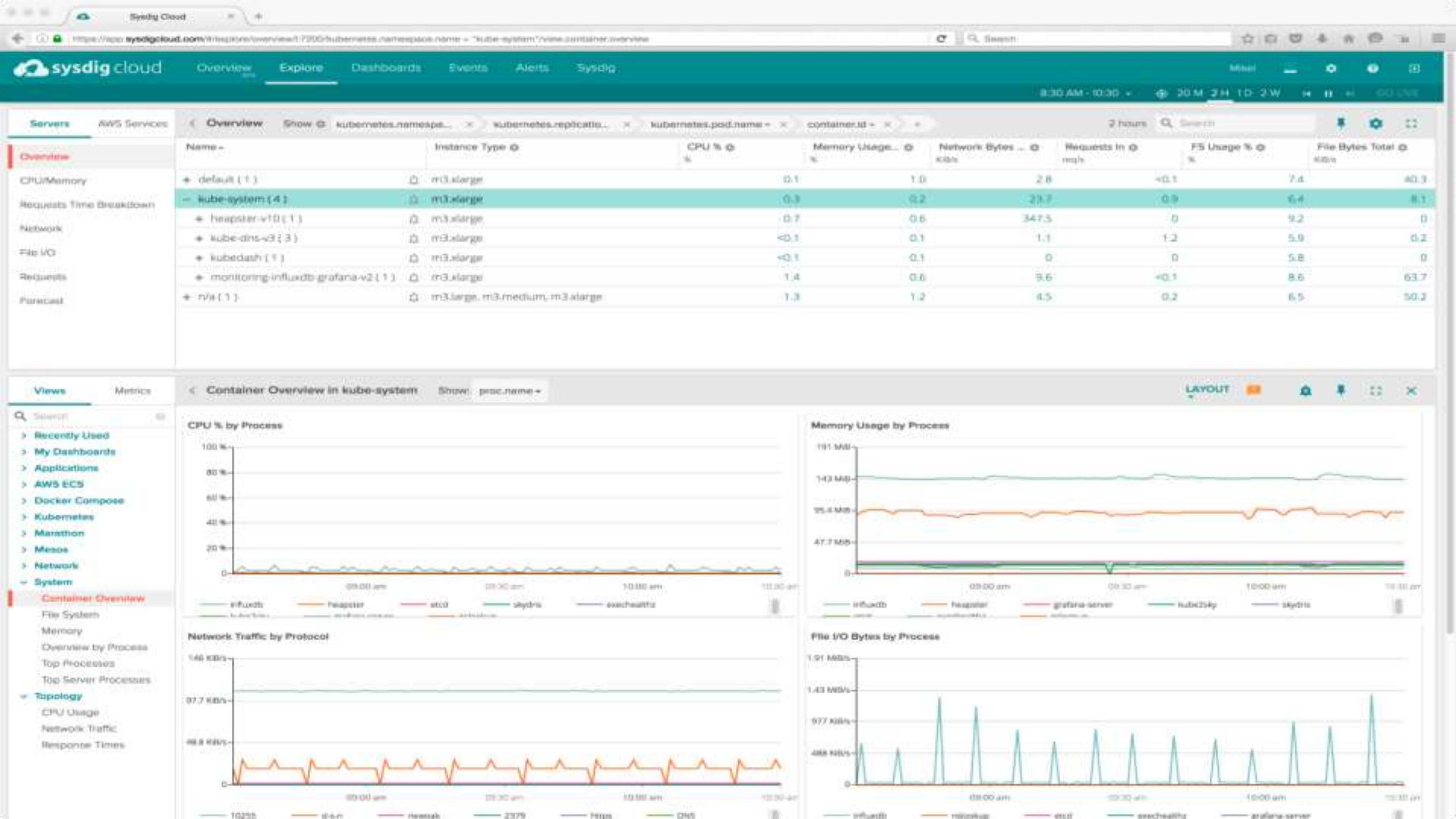
616K lines of code, docs, etc

525 contributors

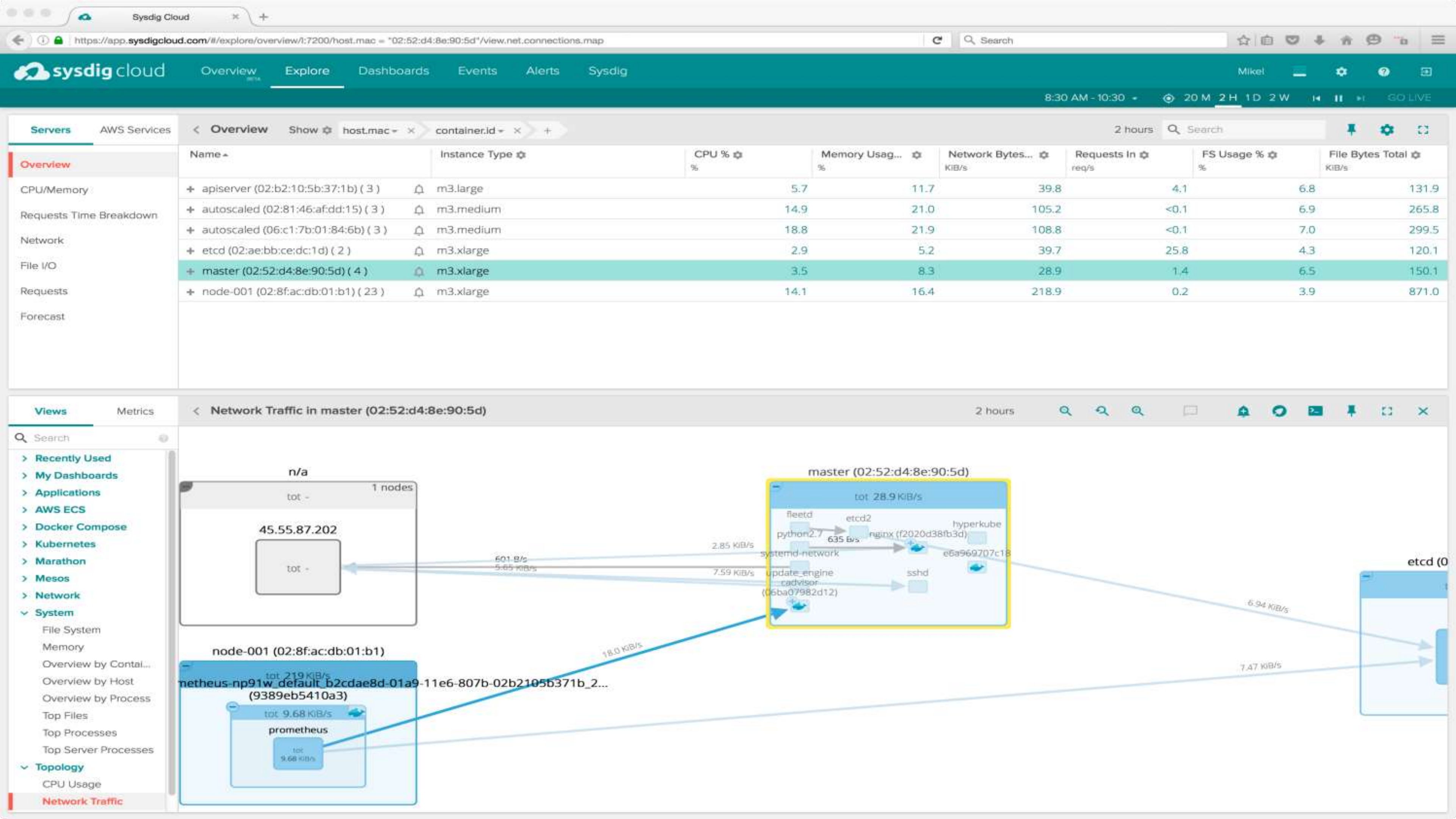
Written in GoLang

What Else?

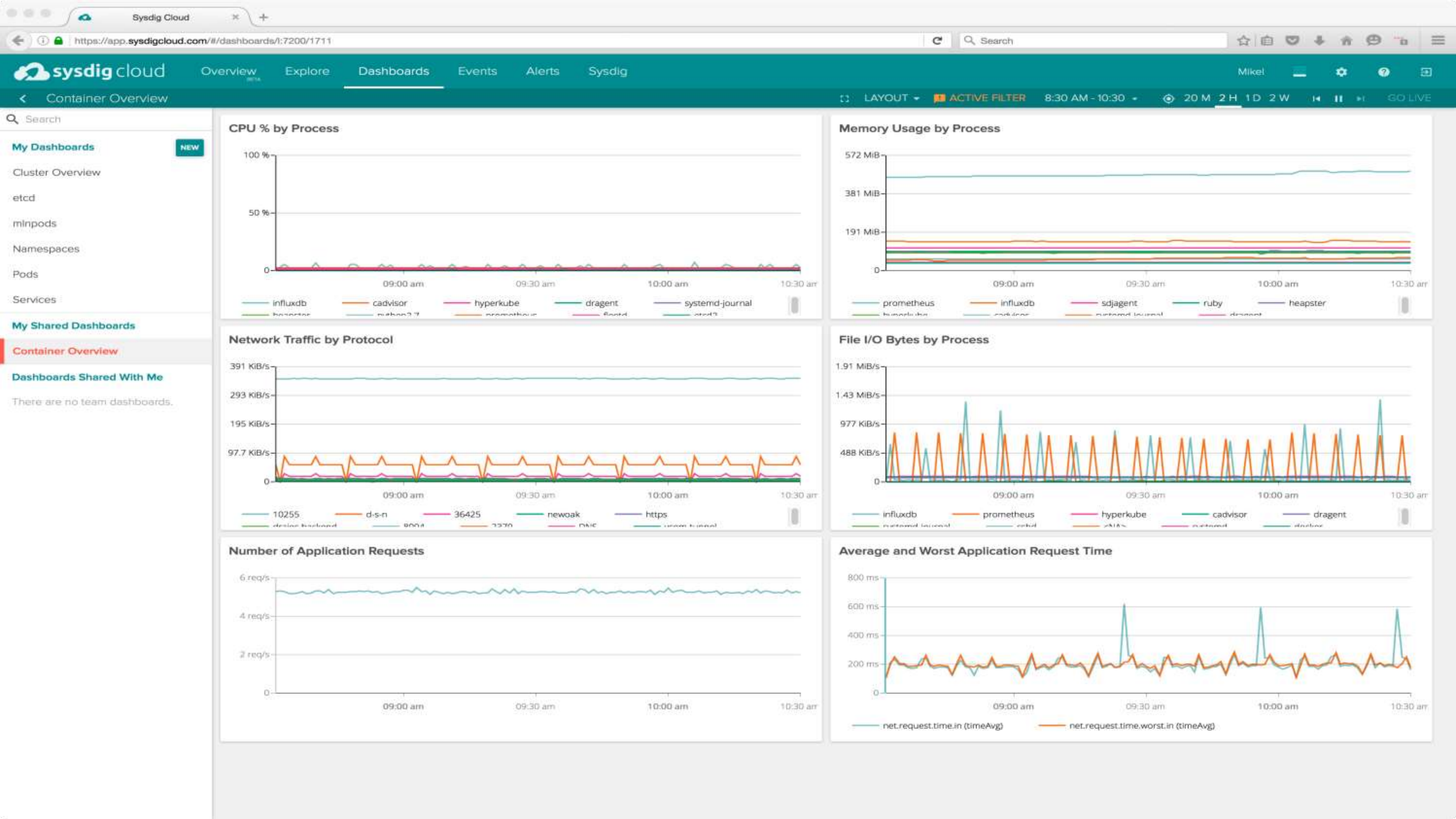
- Orchestration systems like Kubernetes have tooling that supports
  - Integrated load balancers. You simply describe your service endpoint and the internal connections.
  - "Canary" upgrades. (Roll-forward a little bit at a time, roll-backward)
  - Secrets & config info injection
- There's also an ecosystem of technologies that help with deployments, e.g.
  - Sysdig: Kubernetes-aware monitoring



Name	Instance Type	CPU % %	Memory Usage... %	Network Bytes ... KiB/s	Requests In req/s	FS Usage % %	File Bytes Total KiB/s
+ default (1)	m3.xlarge	0.1	1.0	2.8	<0.1	7.4	40.3
− kube-system (4)	m3.xlarge	0.3	0.2	23.7	0.9	6.4	8.1
+ heapster-v10 (1)	m3.xlarge	0.7	0.6	347.5	0	9.2	0
+ kube-dns-v3 (3)	m3.xlarge	<0.1	0.1	1.1	1.2	5.9	0.2
+ kubedash (1)	m3.xlarge	<0.1	0.1	0	0	5.8	0
+ monitoring-influxdb-grafana-v2 (1)	m3.xlarge	1.4	0.6	9.6	<0.1	8.6	63.7
+ n/a (1)	m3.large, m3.medium, m3.xlarge	1.3	1.2	4.5	0.2	6.5	50.2



Name	Instance Type	CPU % %	Memory Usag... %	Network Bytes... KiB/s	Requests In req/s	FS Usage % %	File Bytes Total KiB/s
+ apiserver (02:b2:10:5b:37:1b) (3)	m3.large	5.7	11.7	39.8	4.1	6.8	131.9
+ autoscaled (02:81:46:af:dd:15) (3)	m3.medium	14.9	21.0	105.2	<0.1	6.9	265.8
+ autoscaled (06:c1:7b:01:84:6b) (3)	m3.medium	18.8	21.9	108.8	<0.1	7.0	299.5
+ etcd (02:ae:bb:ce:dc:1d) (2)	m3.xlarge	2.9	5.2	39.7	25.8	4.3	120.1
+ master (02:52:d4:8e:90:5d) (4)	m3.xlarge	3.5	8.3	28.9	1.4	6.5	150.1
+ node-001 (02:8f:ac:db:01:b1) (23)	m3.xlarge	14.1	16.4	218.9	0.2	3.9	871.0



CPU % by Process

Memory Usage by Process

Network Traffic by Protocol

File I/O Bytes by Process

Number of Application Requests

Average and Worst Application Request Time

800 ms
600 ms
400 ms
200 ms
0
09:00 am
09:30 am
10:00 am
10:30 am
net.request.time.in (timeAvg)
net.request.time.worst.in (timeAvg)

A Few Words About PaaS

- Old PaaS integrated its own container and orchestration systems
  - Bad: segregated clusters
  - Bad: Old PaaS is good for initial dev, but a huge threshold to leave for “normal” infrastructure
- Systems like DEIS and Openshift represent new PaaS:
  - Complementary to Kubernetes: adds functionality, doesn’t replace it
- Helm: A way to package a service so that it can be easily deployed by others.

OPENSHIFT by Red Hat

DEIS

A project by engineyard

What Else Is Out There?

- Mesos (Mesosphere). Recent funding from MSFT and HPE. Just open sourced more of their "secret sauce."
- Docker Swarm. Rapid deployment tool (dev-focused), with roadmap for more.
- Amazon, others.
- Emerging market, so will shake out, but:
  - Kubernetes is a multi-source, foundation-based open source project.
  - What we've bet on due to strong support from Google and others.

Samsung SDS & Kubernetes

- Focused team working on cloud native computing
- We are pushing back our work either into Kubernetes or into our github repo – Samsung-AG.
- Very active in the community; leading the Kubernetes Scaling and Testing SIGs.
- Providing professional services to companies adopting cloud native, or to those building products for the cloud native market.

Cloud-Native Reference Stack

Ready-to-Deploy Full (v)DC Stack

www.github.com/samsung-cnct

App	Cassandra	
Orchestration	Kubernetes 1.1 → 1.2	
Networking	Flannel…	
Container	Docker	
Optimized OS	CoreOS	
Compute	AWS, GCE	Metal
Provisioning	Ansible	Rebar
	Terraform	

Stack changes based on demands and evolving technologies

Sysdig, Swift, Calico, … All under evaluation for inclusion

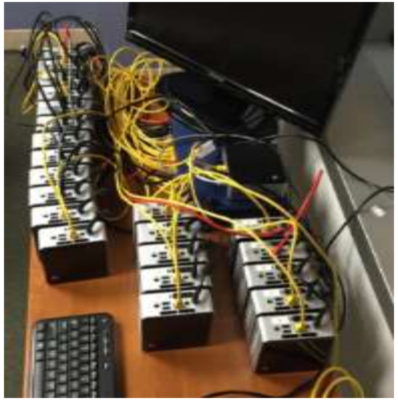
kubernetes

docker

flannel

CoreOS

digital REBAR reinforced datacenter



amazon web services™

SDS Cloud Native Computing Team

World Class Expertise in Modern Best Practices:
Org Design, Automation, Devops, Containers,
Data Centers, Open Source

Seattle, San Diego, San Jose

Q & A

and THANK YOU for your time.

You can find links to this presentation and the demo here.

Richard Kaufmann & Bob Wise
rskaufmann@samsung.com &
bob.wise@samsung.com

www.SDC2016.com

SAMSUNG DEVELOPER CONFERENCE

https://github.com/samsung-cnct